

Traffic Classification over Tor Network Based on RGB Images

Depeng Chen^{1,2}, Xiao Wu¹, Jie Cui^{1,2} (✉), and Hong Zhong¹

¹ School of Computer Science and Technology, Anhui University, Hefei 230601, China

² Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, depengchen, cuijie, zhongh@ahu.edu.cn, e22201128@stu.ahu.edu.cn

Abstract. The emergence of the Tor anonymous communication system effectively protects users' identities from untrusted destinations and third parties on the Internet. However, cases of anonymous abuse, where Tor is used to hide identities and commit cybercrime, are common. Therefore, identifying and researching Tor traffic is particularly important. In this study, we propose an RGB image data processing method combined with deep learning to classify Tor traffic. First, we examine the influence of different image saving formats on model performance. Subsequently, we discuss the impact of various assignment methods for RGB images on experimental results. Finally, we compare the performance of the model trained using the RGB image method with that of the conventional grayscale image method. The experimental results demonstrate that our method effectively identifies different types of Tor traffic.

Keywords: Tor traffic · RGB image · Traffic classification · Deep learning

1 Introduction

Tor [1] was first developed by a technical research laboratory of the U.S. Navy. Through this set of encrypted routing mechanisms, users and website IPs are hidden to prevent being tracked. Because this mechanism encrypts network data packets layer by layer using the cryptographic algorithms, and decrypts them layer by layer on the transmission path to obtain the original data packets [2]. Nowadays, network complexity is increasing, and extracting valuable information from network traffic is crucial. Identifying dark network traffic, especially Tor traffic, is essential for network security. Strengthening network supervision and combating illegal activities online requires effective identification and tracking of Tor traffic.

Identification of Tor traffic can be traced back to 1996. Wagner and Schneier [3] first noticed that packet length characteristics leaked information about traffic data. Subsequently, the researchers successively analyzed each TCP connection, extracted various resource length characteristics of the Web page, and achieved a high accuracy rate through simple statistical calculation methods. With the development of website fingerprint defense technology, the resource length feature is no longer effective. To this end, researchers seek new package metadata features and use machine learning methods to conduct identification of traffic studies. Herrmann et al. [4] applied website

fingerprinting technology to the Tor anonymous network for the first time, using the frequency distribution of packet length as the website fingerprint feature and naive Bayes as the classifier.

In this paper, our main contributions are mainly in three aspects. Firstly, we compared the effects of various image preservation formats on the experimental results to determine the best format. Next, we proposed two multi-channel image assignment methods and discussed their impact on model performance. Finally, we compared the experimental results of the proposed RGB image method with the traditional grayscale image method to verify its effectiveness.

2 Related Work

There are two different classification granularities for the Tor anonymous traffic classification problem: website identification and user behavior identification. Different anonymous traffic classification scenarios require different classification granules. If the network regulatory department aims to monitor the websites visited by Tor users or identify the users' server-side IP addresses, the classification granularity should focus on website identification. If the network supervision department wants to detect which category of Tor traffic a Tor user accesses, the classification granularity is user behavior identification, such as email, video, audio, chat, and other categories of traffic. Our work mainly studies user behavior recognition.

User behavior identification is also called identifying traffic categories. Bai et al. [5] proposed a fingerprint method to identify Tor and Web-Mix networks. Their method uses specific strings, packet lengths, and packet frequencies as features. Their method was tested on a simulated network, and the accuracy of both the Tor system and the Web-Mix system was over 95%. AlSabah et al. [6] proposed a QoS mechanism to improve the differentiation of batch transmission traffic, interactive traffic, and streaming media traffic in the Tor network. They use Tor link lifetime, data transfer, cell arrival time, and number of recently sent cells as features. The accuracy rate exceeds 90% in artificial data sets and 77% in real experiments. He et al. [7] proposed a hidden Markov model method to classify encrypted Tor traffic into 4 categories: P2P, FTP, IM, and Web. They used the volume and direction of data extracted from Tor streams as features and ultimately achieved an accuracy of 92%. Lashkari et al. [8] proposed a Tor traffic feature extraction method based on time features and accessed different servers through the Whonix operating system to collect real-time Tor traffic data from eight categories of traffic, including Browsing, Email, Chat, Audio Streaming, Video Streaming and other, between the client and server. Finally, the corresponding time features are extracted from the collected data and classified using different machine learning methods to achieve relatively high accuracy.

In addition to machine learning algorithms being used for user behavior recognition, as deep learning methods have achieved good results in fields such as computer vision and speech recognition, researchers are trying to apply deep learning methods to the study of user behavior recognition. Zhou et al. [9] used the normalization method to process the traffic data and mapped it into a gray image as the input data of the

convolutional neural network. Then, the parameters of the feature map and the fully connected layer were processed. Designed to select the optimal classification model to implement traffic classification. Lin et al. [10] proposed that the traffic feature vector can be converted into a feature matrix to form the input image of CNN. Yu et al. [11] constructed HTTP traffic as a natural language sequence and proposed a method to detect abnormal HTTP traffic based on bidirectional LSTM. Wang et al. [12] pointed out that network traffic can be converted into gray images, and then CNN can be used for training, learning, and classification recognition.

The above research shows that deep learning has great potential in Tor traffic classification. Deep learning algorithms do not need to perform complicated feature engineering on input data, but can autonomously mine relevant abstract features from a large amount of raw data. Compared with traditional machine learning algorithms, classification models based on deep learning have higher accuracy.

This paper proposes a new Tor traffic classification method based on RGB images. It intercepts the first part of the bytes of each data packet and maps them into the form of RGB color images thereby retaining the most completed raw data. Then a traffic classification model that can autonomously learn features is constructed using the LeNet-5 network [13]. The optimal classification effectiveness has been achieved compared to previous Tor traffic classification technologies.

3 Tor Traffic Classification Method Based on RGB Images

This section will introduce in detail the Tor traffic classification method based on RGB images proposed in our paper, including a CNN training model used, the processing method of Tor network traffic data, and the experimental settings. As shown in Fig. 1, the Tor traffic classification process based on RGB images consists of three parts: traffic preprocessing module, CNN model training module, and CNN model testing module.

3.1 Convolutional Neural Network

Convolutional neural network is a type of deep learning algorithm used to process data with matrix structure. It has achieved remarkable success in computer vision tasks and is widely used in fields such as image recognition and target detection. Its structure usually consists of input layer, convolution layer, pooling layer, fully connected layer and output layer.

Input layer. The input layer is used for data input. It converts image data into a pixel matrix and performs some preprocessing operations. Several common image preprocessing methods include scaling, normalization and standardization.

Convolution layer. The main job of the convolution layer is to perform a convolution operation on the convolution kernel and the image to obtain a new feature map. It uses a set of learnable convolution kernels to perform convolution operations on the input

image and extract local features. Through convolution operations, convolutional layers can capture spatial local relationships and invariance in images.

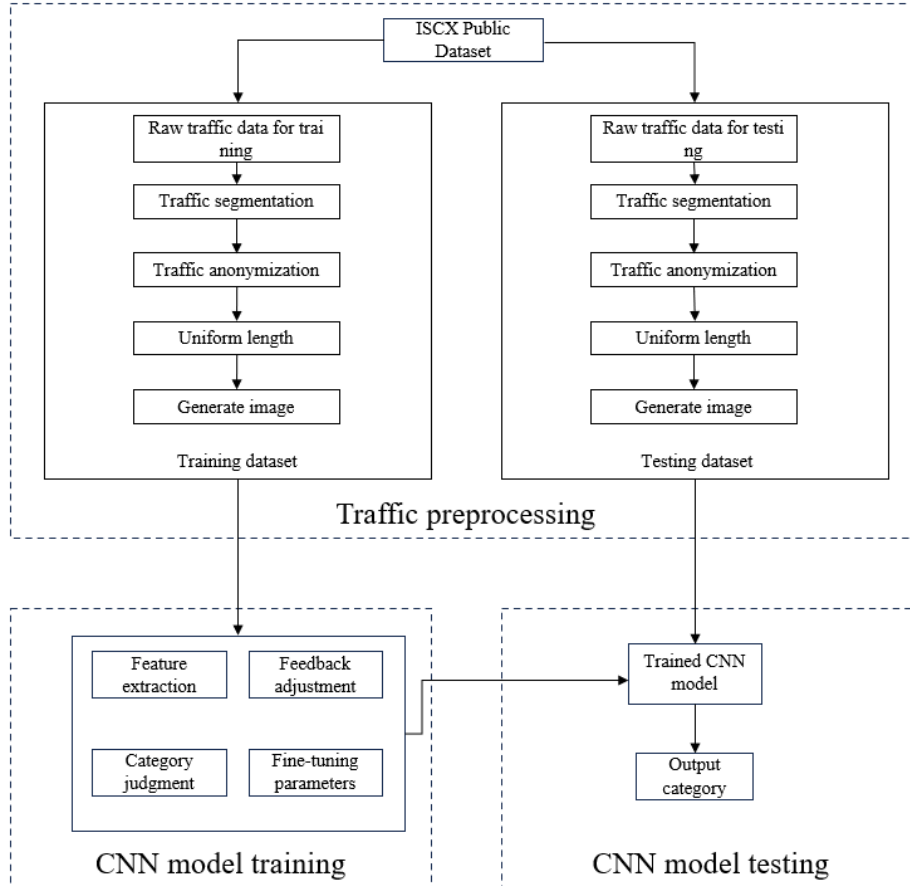


Fig. 1. Flowchart for Tor traffic classification using RGB images.

The calculation method is to scan the image with a convolution kernel according to a certain step size, and perform multiplication and addition operations on all corresponding elements within each scan. This process yields a new feature map. Typically, to obtain multiple feature maps, the convolution layer may employ multiple convolution kernels, each performing a separate convolution operation to generate a new feature map. As convolution remains a linear operation, an activation function is applied to perform a nonlinear mapping of the convolution result. Commonly used activation functions include sigmoid, ReLU, and tanh functions.

Pooling layer. The pooling layer is used to reduce the spatial size of the feature map and the number of parameters. Common pooling operations include max pooling and

average pooling. Through the pooling operation, the spatial dimension of the feature map can be reduced, important feature information can be retained, and the calculation of the network can be simplified.

Fully connected layer and output layer. Fully connected layers typically consist of a large number of parameters and are utilized to learn complex relationships within data. These layers integrate and transform features extracted by preceding convolutional and pooling layers, transmitting the output values to the classifier. Conversely, the output layer is responsible for producing the final target result.

3.2 CNN Architecture Modification

LeNet-5 is the first convolutional neural network model to achieve an important breakthrough in handwritten digit recognition. The training model used in this paper is mainly based on the traditional LeNet-5 structure [13], and the following improvements have been made: (1) The input size of LeNet-5 was adjusted to accommodate the classification of Tor traffic. (2) As our paper aims to differentiate between various types of Tor traffic and regular traffic, totaling 9 categories, the output of the LeNet-5 model was adjusted from 10 neurons to 9 neurons, representing each category of traffic. The convolutional neural network utilized in our paper has its specific structure, as depicted in Fig. 2.

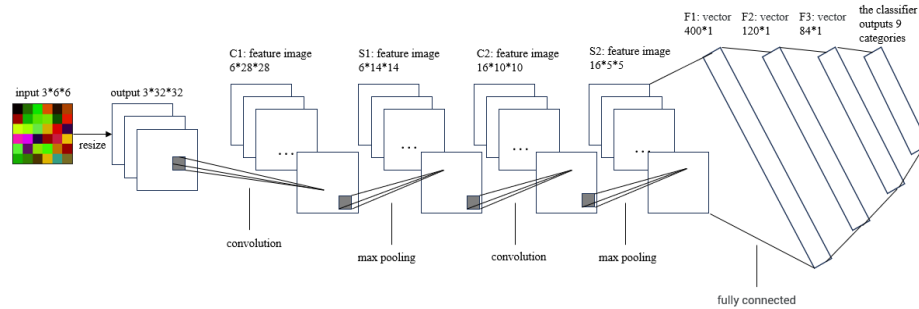


Fig. 2. CNN structure for Tor traffic classification.

3.3 Tor Traffic Data Preprocessing Method

Prior to training a convolutional neural network model using Tor traffic data, it is necessary to preprocess the Tor traffic data.

Traffic segmentation. Raw traffic data is typically saved in the pcap file format, which can be opened using Wireshark software to inspect the data packets. A traffic data file usually contains thousands of data packets, with each packet containing all the information of the traffic data. Thus, the initial step of this method involves segmenting the traffic data file into units of data packets to facilitate processing.

Traffic anonymization. To ensure that the final classification is not influenced by MAC address and IP address, regardless of the traffic data packet's category, this method anonymizes the network traffic. Specifically, it replaces the source MAC address, destination MAC address, source IP address, and destination IP address with hexadecimal 0x00, totaling 20 bytes.

Uniform length. To facilitate training and classification, it's necessary to standardize the input size of the model due to the varying lengths of different data packets. Based on our observation, Tor traffic typically consists of: 14 bytes of Ethernet header, 20 to 60 bytes of IP header, 20 to 60 bytes of TCP/UDP header, 0 to 1480 bytes of application layer data, and 4 bytes of Ethernet tail. Since encrypted traffic often encrypts the application layer data, rendering it ruleless and unsuitable for feature learning, our paper concludes that the neural network model cannot effectively learn from it for classification of encrypted network traffic. Additionally, the 4-byte Ethernet tail is used for error checking during transmission and is irrelevant to traffic categorization, so it is also removed. Consequently, after removing the application layer data and Ethernet tail, only the Ethernet header, IP header, and TCP/UDP header remain. The minimum size of this portion of traffic data is 54 bytes, and the maximum is 134 bytes. Further observation reveals that only a very small portion of traffic includes IP and TCP optional fields. Therefore, our paper sets the input size to $N=100$ bytes or $N=108$ bytes, with any remaining space filled with zeros.

Generate image. The decimal values corresponding to the first 100 bytes of each data packet are converted into gray values ranging from 0 to 255 pixels. Subsequently, each data packet is transformed into a 10×10 gray image, which serves as the volumetric input to the neural network. Additionally, the first 108 bytes of each data packet can be converted into three separate images, each ranging from 0 to 255. These three images represent the red, green, and blue channels of an RGB image. Finally, the three images are combined into a multi-channel RGB picture, which is then utilized as the input of the convolutional neural network.

4 Experiment

This section will introduce the specific process and details of the experiment, including the experimental environment, datasets, evaluation methods, and experimental steps.

4.1 Experimental Environment Setup

The experimental environment configuration is shown in Table 1.

4.2 Experimental Dataset Classification

The experimental samples used the ISCX¹ dataset publicly available online. The dataset designers captured outgoing traffic from workstations and gateways, collecting a set of pcap files containing both regular traffic and Tor traffic. This experiment specifically selects 7 categories of Tor traffic for analysis.

Table 1. Experimental environment setup.

Category	Parameters
CPU	AMD R9 7940H
Graphics card	NVIDIA RTX 4060
Operating system	Windows 10 Home Chinese version
Python interpreter	Python 3.9.12
Python editor	PyCharm 2023.1
Neural network building platform	Pytorch 2.0.0

- 1) Tor Browsing (tor_browsing): Accessing various web pages.
- 2) Tor Chat (tor_chat): Chat data from ICQ, AIM, Skype, Hangouts, and Facebook.
- 3) Tor Mail (tor_mail): Emails encapsulated by SMTPS, POP3S, and IMAPS protocols.
- 4) Tor File Transfer (tor_file): File transfers via Skype, FTP, and other protocols.
- 5) Tor P2P File Transfer (tor_p2p): uTorrent and BitTorrent file transfers.
- 6) Tor VoIP (tor_voip): Voice calls via Skype, Facebook, and Hangouts.
- 7) Tor Video Playback (tor_video): Video playback from YouTube and Vimeo.

Since non-Tor traffic types in the ISCX2016 public dataset are unclear, the experiment focuses on two major categories of non-Tor traffic:

- 8) Browsing: Accessing various web pages.
- 9) Chat: Chat data from ICQ, AIM, Skype, Hangouts, and Facebook.

For the above data, training and testing datasets are constructed with a ratio of 3:1. The training dataset (train_data) consists of 900 images from each category, totaling 8100 images, for training the neural network. The test dataset (test_data) comprises 300 images from each category, totaling 2700 images, to evaluate the classification performance of the trained neural network.

4.3 Experimental Evaluation Criteria

This experiment uses Accuracy, Precision, Recall and F1 value for performance testing. These four indicators are also the most commonly used indicators in deep learning. The calculation formulas for these four indicators are as follows:

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad (1)$$

¹ <https://www.unb.ca/cic/datasets/tor.html>

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

$$F1 = \frac{2*(Precision*Recall)}{(Precision+Recall)} \quad (4)$$

4.4 Experimental Results

We conducted three main experiments: 1) Comparing the impact of image-saving formats on experimental results; 2) Analyzing the impact of different assignment methods on experimental results; 3) Comparing the effects of gray images and RGB images.

The impact of image saving formats. There are many types of images saving formats. Our paper mainly discusses images saved in JPG and PNG format. The left and right images in Fig. 3 represent two different storage formats, JPG and PNG, respectively, obtained by converting the first 108 bytes of the same data packet into a multi-channel RGB image. The difference between the two images in different formats is clearly visible. Hence, the choice of saving format for RGB images is crucial to the performance of the trained model.

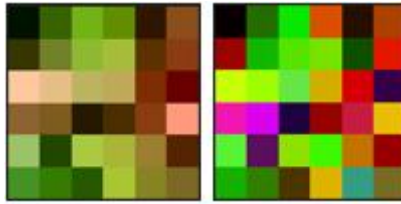


Fig. 3. JPG and PNG format images.

In this regard, our paper only changes the image saving format in the data preprocessing stage while ensuring that the original data set remains unchanged. As shown in Table 2, it can be seen that the accuracy of the model with images saved in PNG format is significantly higher than the accuracy of the model with images saved in JPG format.

We explain this difference accordingly. The JPG format uses lossy compression to reduce file size, which means that some image information will be lost during the process of saving the image, especially details that have little visual impact. The PNG format uses lossless compression and can retain all image data. This also directly leads to the difference between the left and right images of Fig. 3, which in turn leads to different experimental results. Therefore, the subsequent experiments in our paper will save the images in PNG format.

Table 2. Comparison of model accuracy under different saving formats.

number of times	model accuracy under JPG format	model accuracy under PNG format
first time	97.51%	98.96%
second time	97.21%	98.65%
third time	97.49%	98.80%
fourth time	97.20%	98.67%
fifth time	97.35%	98.87%

RGB image assignment method. Compared with gray images, the assignment methods for multi-channel images are more diverse, so this experiment explores the impact of assignment methods on model performance.

Assigning bytes from a data packet to pixels in a gray image is straightforward since it's a single channel. However, for RGB images, the assignment order is crucial.

Scheme 1: Assign the R channel first, followed by the G channel, and finally the B channel. Each channel is assigned a value based on the gray image, resulting in a synthesized RGB image.

Scheme 2: Since each pixel in an RGB image consists of values from three color channels, each channel's value can be assigned synchronously. Then, pixels are assigned in a left-to-right and top-to-bottom manner from the gray image. Finally, we synthesize an RGB image.

Additionally, based on these two assignment methods, the order of assignment for each channel can be adjusted. For example, one can assign the R channel first, then the G channel, and finally the B channel, or vice versa. There are a total of 6 arrangements to choose from. As illustrated in Fig. 4 to Fig. 5, adjusting the order of assignment according to these two methods results in 12 different arrangements for the same data packet.

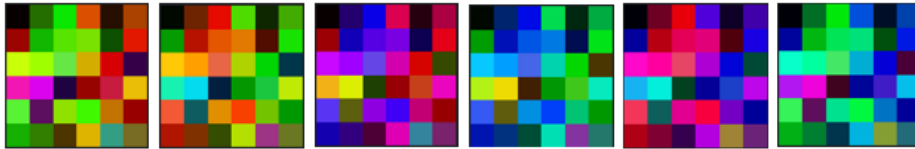
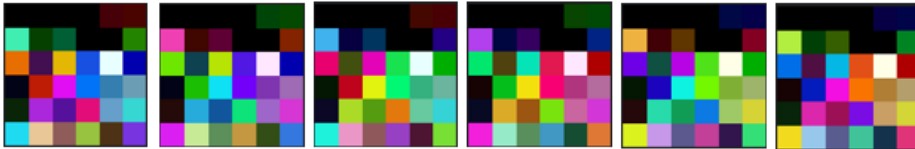
**Fig. 4.** Six assignment orders for scheme 1.**Fig. 5.** Six assignment orders for scheme 2.

Fig. 6 is a comparison of the model accuracy of two assignment methods and adjusting different assignment orders. It can be seen from the data that adjusting the order of

assignments in the same assignment method has little impact on the accuracy of the model. This process is akin to adjusting the order of weights for each channel in a convolutional neural network, thus it may not significantly impact experimental results. Interestingly, for the two different assignment schemes, scheme 1 consistently yields higher accuracy compared to scheme 2. This discrepancy could be attributed to the presence of a certain order relationship between bytes. Dividing continuous bytes into different picture channels may disrupt this order relationship, resulting in a decrease in accuracy.

Two methods of data preprocessing. This experiment investigates the influence of gray image data preprocessing methods and RGB image data preprocessing methods on model performance.

A gray image is a representation containing only gray information. Each pixel's value signifies its brightness or gray level, typically ranging from 0 to 255, where 0 represents black and 255 represents white. Gray images are single-channel images, with each pixel having only one value. On the other hand, an RGB image is based on three color channels: red, green, and blue. Each pixel comprises values from these three channels, determining the intensity of red, green, and blue respectively. The pixel value of each channel in an RGB image also falls between 0 and 255. Combining the pixel values of different channels yields a color image. Compared to gray images, RGB images can represent richer color and more detailed information.

According to the data preprocessing method outlined in section 3.3, we convert the decimal values corresponding to the first 100 bytes of each data packet into a 10*10 gray image. Additionally, we convert the decimal values corresponding to the first 108 bytes of each data packet into a 6*6*3 multi-channel RGB picture. The performance of the network is evaluated based on precision, recall, F1 value for each category, and overall classification accuracy.

Fig 7 illustrates the evolution of model accuracy with increasing training rounds. It is evident that the model's accuracy stabilizes after approximately ten rounds of training, with the RGB image method consistently outperforming the gray image method. To mitigate experimental data variability, we modified the assignment order of RGB images and compared six different assignment orders with the gray image method.

Table 3 and Table 4 evaluate the model based on precision, recall rate, and F1 value. According to the test results of each indicator, our proposed RGB data preprocessing method outperforms the gray image data preprocessing method. Therefore, the effectiveness of our proposed RGB image data preprocessing method is confirmed.

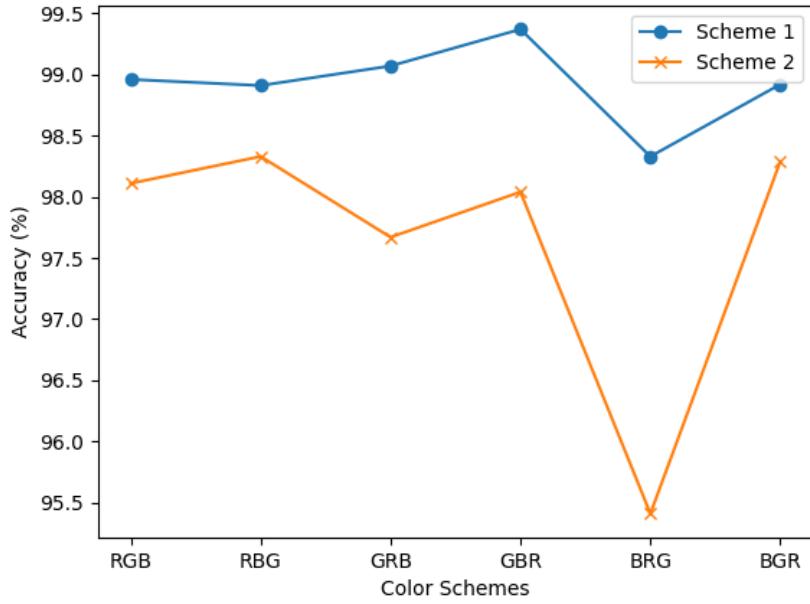


Fig. 6. Comparison of accuracy rates of different assignment methods.

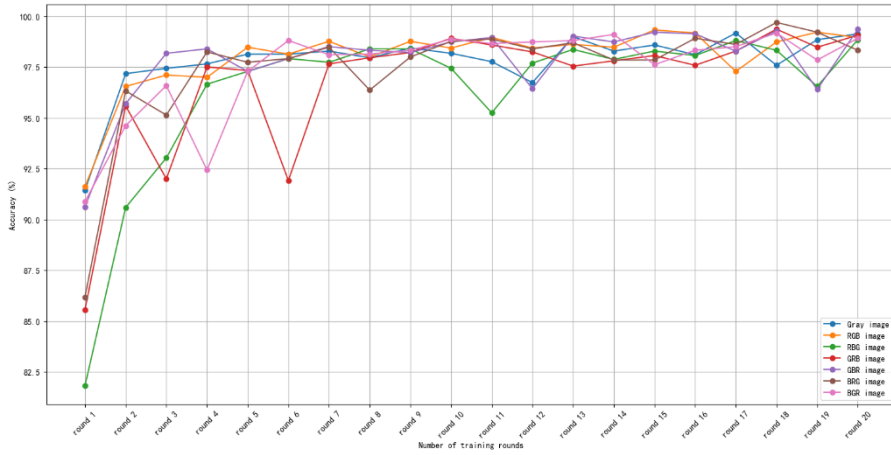


Fig. 7. Comparison of accuracy under different assignment orders and training rounds.

Table 3. Performance testing of the model using the gray image method.

Category	Precision	Recall	F1
tor_browsing	99.668%	100.000%	99.834%
tor_chat	99.664%	99.000%	99.331%
tor_file	99.007%	99.667%	99.336%
tor_mail	100.000%	100.000%	100.000%
tor_voip	100.000%	99.333%	99.666%
tor_p2p	99.668%	100.000%	99.834%
tor_video	100.000%	100.000%	100.000%
browsing	97.966%	96.333%	97.143%
chat	96.393%	98.000%	97.190%
mean	99.151%	99.148%	99.148%

Table 4. Performance testing of the model using the RGB image method.

Category	Precision	Recall	F1
tor_browsing	99.668%	100.000%	99.834%
tor_chat	100.000%	98.667%	99.329%
tor_file	99.010%	100.000%	99.502%
tor_mail	99.338%	100.000%	99.668%
tor_voip	100.000%	100.000%	100.000%
tor_p2p	100.000%	100.000%	100.000%
tor_video	100.000%	100.000%	100.000%
browsing	98.328%	98.000%	98.164%
chat	97.993%	97.667%	97.830%
mean	99.370%	99.370%	99.369%

5 Conclusion

In this paper, we introduce a Tor traffic classification method based on RGB images. Our approach involves preprocessing the traffic dataset to generate RGB images, followed by selecting a suitable convolutional neural network model for training. To assess the effectiveness of our proposed method, we compare it with the traditional gray image preprocessing method, demonstrating superior accuracy. Additionally, we investigate the impact of different assignment methods of RGB images on model performance, as well as the significance of image-saving formats on experimental results. Future research will concentrate on designing a model capable of identifying unknown categories of traffic. As our collected traffic categories are limited, distinguishing unknown traffic into unknown categories presents a significant research challenge.

Acknowledgments. This work is supported by the University Synergy Innovation Program of Anhui Province (Project No. GXXT-2022-049) and Key Project of Anhui Provincial Department of Education (Project No. 2022AH050075).

References

1. Dingledine, R., Mathewson, N., Syverson, P.F.: Tor: The second-generation onion router. *USENIX security symposium*. **4**, 303-320 (2004)
2. Winter, P., Edmundson, A., Roberts, L.M., et al.: How do tor users interact with onion services? *USENIX Security Symposium*, 411-428 (2018)
3. Wagner, D., Schneier, B.: Analysis of the SSL 3.0 protocol. *The Second USENIX Workshop on Electronic Commerce Proceedings*. **1**(1), 29-40 (1996)
4. Herrmann, D., Wendolsky, R., Federrath, H.: Website fingerprinting: attacking popular privacy enhancing technologies with the multinomial naïve-bayes classifier. *The 2009 ACM workshop on Cloud computing security*, 31-42 (2009)
5. Bai, X., Zhang, Y., Niu, X.: Traffic identification of tor and web-mix. *Eighth International Conference on Intelligent Systems Design and Applications*. **1**, 548-551. IEEE (2008)
6. AlSabah, M., Bauer, K., Goldberg, I.: Enhancing Tor's performance using real-time traffic classification. *The 2012 ACM conference on Computer and communications security*, 73-84 (2012)
7. He, G., Yang, M., Luo, J., et al.: Inferring application type information from tor encrypted traffic. *2014 Second International Conference on Advanced Cloud and Big Data*, 220-227 (2014)
8. Lashkari, A.H., Gil, G.D., Mamun, M.S.I., et al.: Characterization of tor traffic using time based features. *International Conference on Information Systems Security and Privacy*. **2**, 253-262 (2017)
9. Zhou, H., Wang, Y., Lei, X., et al.: A Method of Improved CNN Traffic Classification. In: *2017 13th International Conference on Computational Intelligence and Security (CIS)*, pp. 177-181. IEEE (2017)
10. Lin, W., Lin, H., Wang, P., et al.: Using Convolutional Neural Networks to Network Intrusion Detection for Cyber Threats. In: *2018 International Conference on Applied System Innovation (ICASI)*, pp. 1107-1110. IEEE (2018)
11. Yu, Y., Liu, G., Yan, H., et al.: Attention-based Bi-LSTM Model for Anomalous HTTP Traffic Detection. In: *2018 15th International Conference on Service System and Service Management (ICSSSM)*, pp. 1-6. IEEE (2018)
12. Wang, W., Zhu, M., Zeng, X., et al.: Malware traffic classification using convolutional neural network for representation learning. In: *2017 International conference on information networking (ICOIN)*, pp. 712-717. IEEE (2017)
13. Zhang, J., Yu, X., Lei, X., et al.: A novel deep LeNet-5 convolutional neural network model for image recognition. *Computer Science and Information Systems*. **19**(3), 1463-1480 (2022)